new relic

THE 2026 STATE OF AI CODING REPORT

INDUSTRY INSIGHTS

A New Relic study, conducted by Hanover Research, surveyed technology leaders (N=200) at the manager level and above at organizations using generative or agentic AI for software and application development.

Recent industry benchmarks establish a clear trajectory for the use of AI for software and application development. Microsoft reports that AI now authors 30% of its codebase (CNBC), while Salesforce CEO Marc Benioff estimates AI agents currently handle between 30% and 50% of the workload (Bloomberg). Meta has signaled a target of 50% for 2026 (Mashable). GitHub Copilot's telemetry suggests an average code share of 46% across its user base (Larridin). And perhaps most notably, Google recently announced that 75% of their code is AI-generated (Business Insider).

In a new landmark study, New Relic found that two-thirds of technology leaders surveyed (67%) report that AI now generates or significantly refactors between 51% and 75% of their organization's weekly code output. The median organization is now operating in a codebase that is mostly written by something other than a person.

That shift has consequences that this report works through in detail. Some are positive. Feature delivery is faster, deployment frequency is up, and a clear majority of leaders rate AI-generated code as higher quality than human-authored code.

Some are not. Production incidents, technical debt, microservice sprawl, and senior-engineer rework time have all increased at the same time.

And the four most common production failure modes from AI-generated code (integration failures, compliance issues, data integrity problems, and security vulnerabilities) have each affected roughly three in ten organizations in the last six months.

THE MOST INTERESTING FINDING

The gap between those two pictures

AI-generated code is rated higher quality at the moment it is reviewed, and it produces measurably worse outcomes once it reaches production. That gap is what this report is about, and it is the strategic opening for the observability category in 2026.

KEY FINDINGS

THERE IS A GAP BETWEEN HOW AI CODE IS RATED IN REVIEW AND HOW IT BEHAVES IN PRODUCTION

This is the central contradiction of the AI-assisted era. 94% rate AI-generated code higher quality than human-authored code at review; yet once it ships, 78% report more incidents, 86% report more senior-engineer firefighting, 74% report at least 25% of AI-generated code needs significant rework, and 82% have already hit at least one production failure tied to it in the past six months.

↑ 94% Consider AI code higher quality than human

↓ 78% Report an increase in incidents during production

VIBE CODING IS MAINSTREAM, PRODUCTION-GRADE, AND APPROACHING GOOGLE'S VOLUME CURVE

Two-thirds of organizations now say 51% to 75% of code is AI-generated, and 88% have written vibe coding into formal production policy. Just 5% restrict it to non-production, and zero ban it.

Two-thirds say most of their code is AI-generated

UNVERIFIED TRUST IN AI GENERATION MAY BE DRIVING A DOWNSTREAM PRODUCTION CRISIS

The core contradiction of the vibe coding era may stem from a profound misplacement of trust early in the lifecycle. Nearly two-thirds of technology leaders (62%) report that their teams often trust AI-generated code enough to ship it without line-by-line manual verification. This widespread, unverified acceptance may explain why perceived code quality remains artificially high during initial reviews, yet crumbles in the wild.

62% Disclosed that their teams trust AI code before review

OBSERVABILITY IS NOW ESSENTIAL FOR AI CODE AND TELEMETRY IS BEING PROMPTED INTO THE OUTPUT

96% of leaders rate observability very or extremely important for AI-generated code, and zero rate it slightly or not important. 78% of teams now prompt AI to include telemetry (logs, traces, customer metrics) as part of the generated code itself.

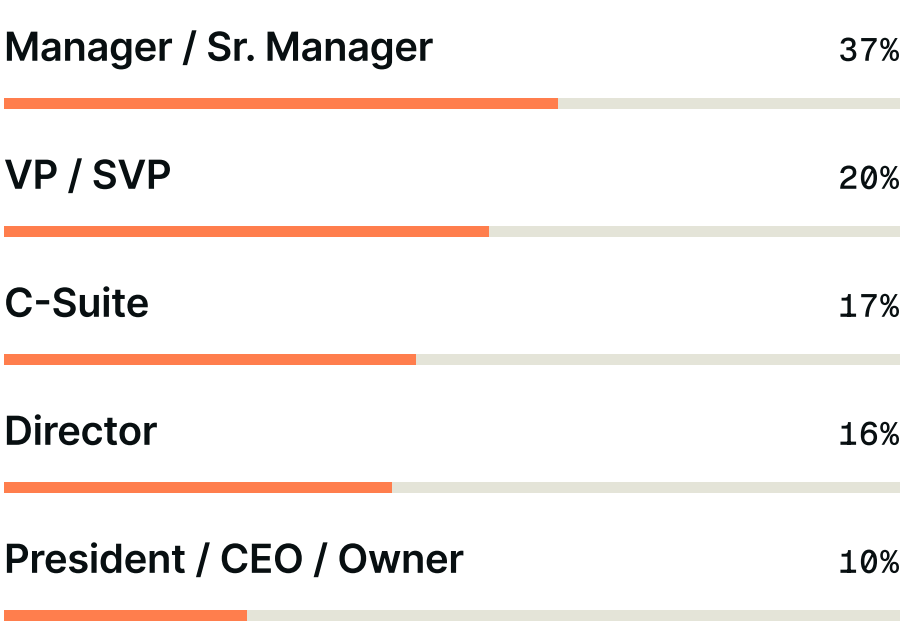
ZERO Rate observability as unimportant for AI code

METHODOLOGY

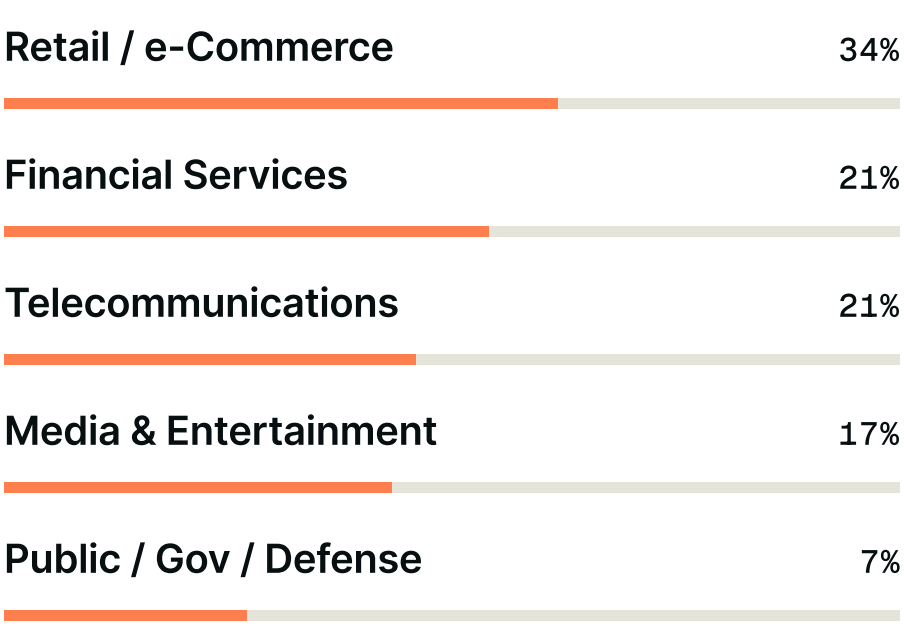
New Relic commissioned a survey conducted online by Hanover Research in 2026. The sample consists of 200 U.S.-based technology decision-makers across IT and engineering, all employed full-time at organizations using generative AI for software development. All respondents are manager level or above and all have meaningful software purchase authority (65% primary decision-maker, 34% shared).

Figures in this report are presented as percentages from the n = 200 base. Headline statistics in the narrative reference the most informative segment of a distribution; charts show the full distribution behind each data point.

SENIORITY OF RESPONDENTS



INDUSTRY COVERAGE



ORGANIZATION REVENUE CONCENTRATES IN THE UPPER MID-MARKET AND ENTERPRISE

92%
ORGS HAVE 500+ EMPLOYEES

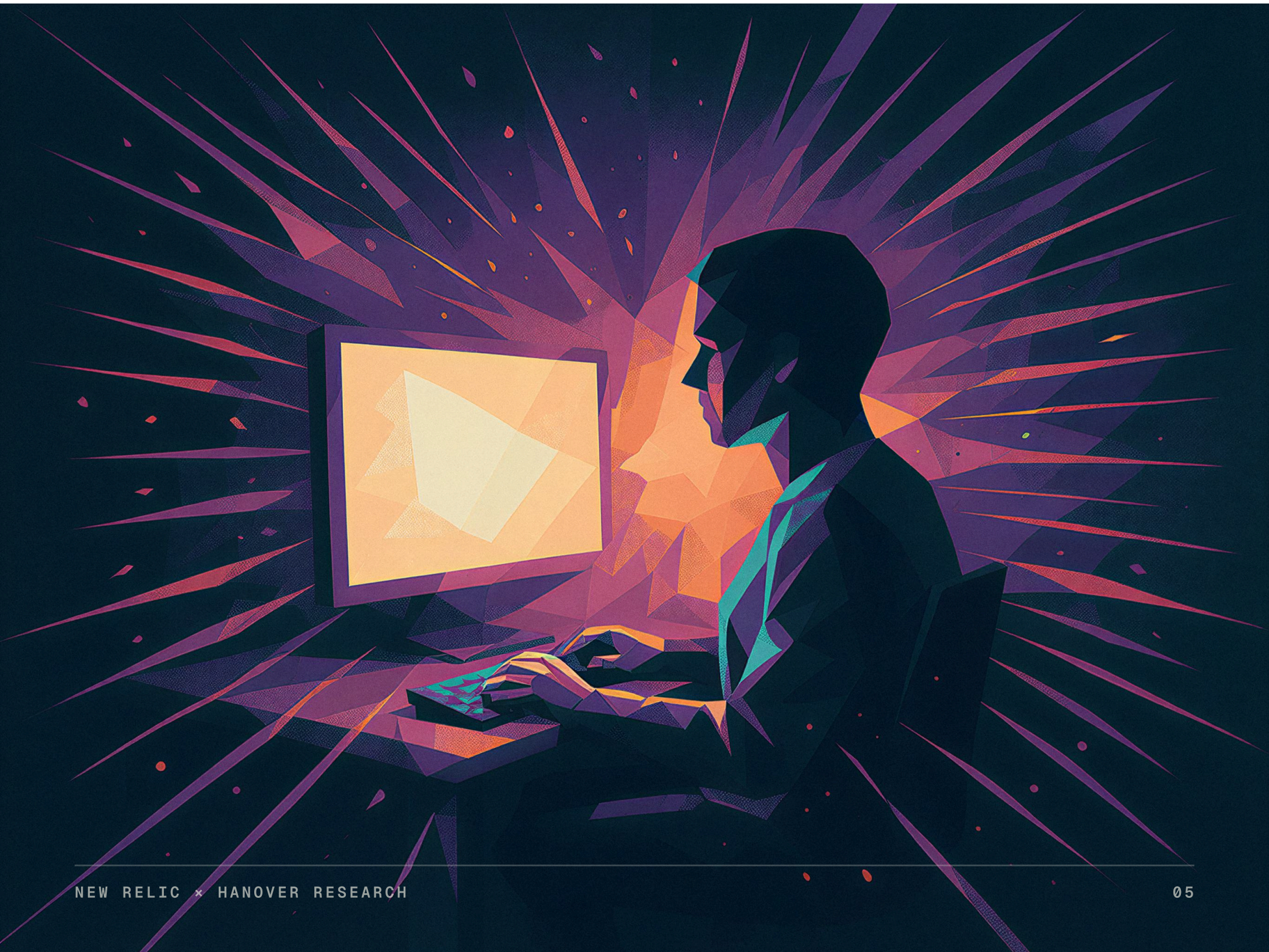
18%
REPORT $250M-$499M ANNUAL REVENUE

37%
REPORT $500M-$999M ANNUAL REVENUE

33%
REPORT $1B+ ANNUAL REVENUE

THE NEW NORMAL

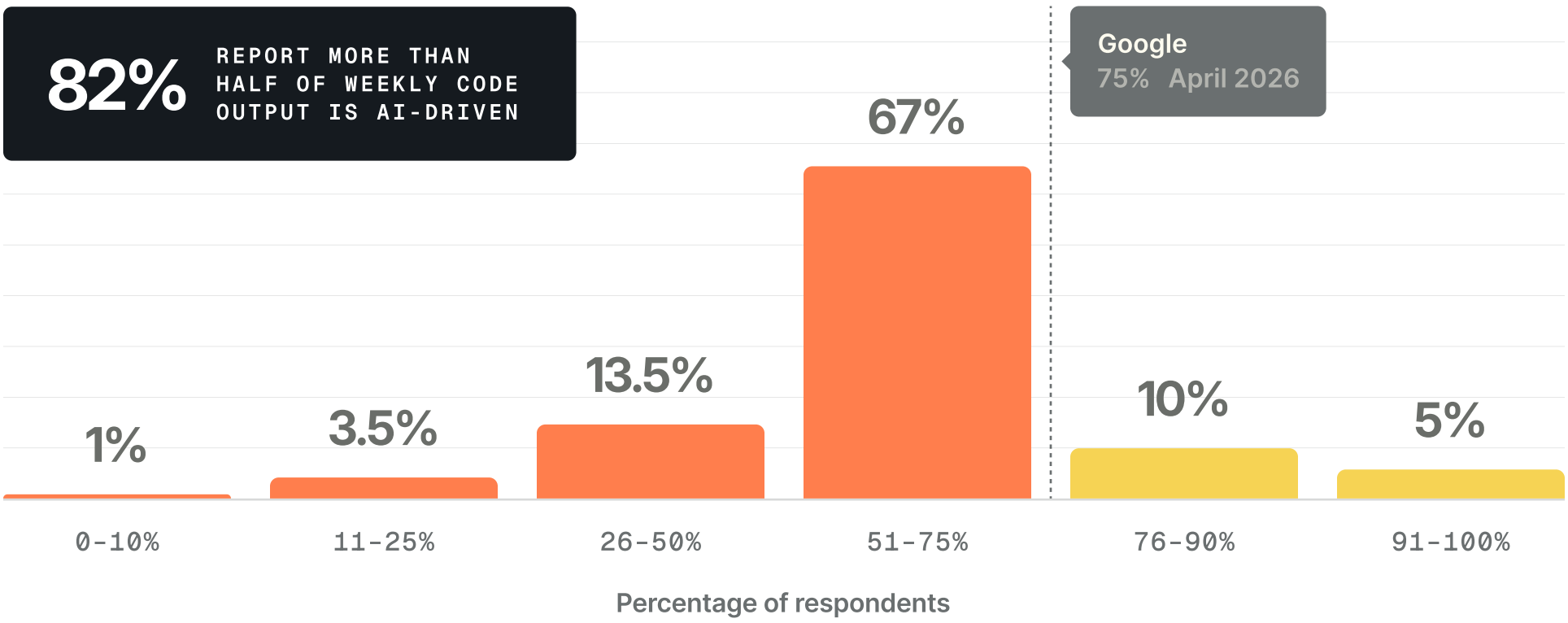
AI-generated code is officially baked into the software development lifecycle rather than being a sandbox hack.



AI-assisted coding has now become the norm

67% of leaders place their organization's weekly AI-generated code output in the 51% to 75% band.

SHARE OF WEEKLY CODE THAT IS AI-GENERATED OR SIGNIFICANTLY REFACTORED BY AI N=200



WHAT IT MEANS

Google's headline 75% number was widely treated as an outlier when it was published. The data in this study reframes it. The 51% to 75% band is now the modal experience for U.S. technology leaders. For most respondents, AI is the primary author and the human is the reviewer.

At these volumes, the engineer responding to a production incident is statistically unlikely to have written the code that caused it. That changes what "understanding the system" means.

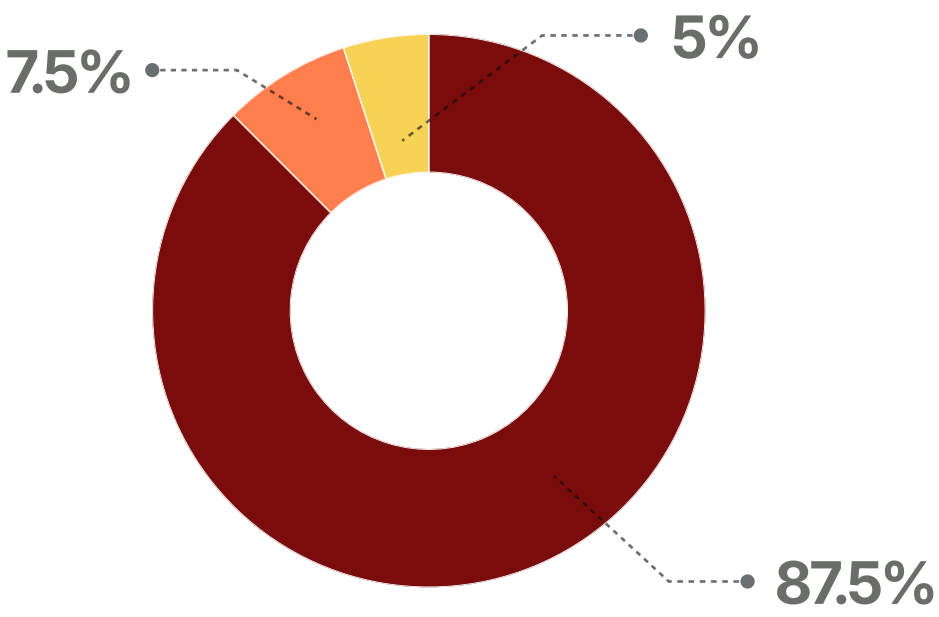
Reading the source is no longer the primary path to comprehension. There is too much of it, generated too fast, by too many different prompts. Runtime evidence (what the code actually did, with what inputs, in what sequence, with what downstream effect) becomes the more reliable signal. Engineering organizations that treat their observability platform as the system of record for "what is actually happening" will scale more cleanly than those that still rely on code review as the comprehension layer.

Vibe coding has cleared the production policy bar

Just 5% of organizations restrict vibe coding to non-production work, and zero ban it outright. The other 95% permit it for production services, either formally (87.5%) or informally (7.5%).

ORGANIZATIONAL POLICY ON VIBE CODING

N=200



95% PERMIT VIBE CODING IN PRODUCTION

- OFFICIALLY PERMITTED IN PRODUCTION
- INFORMALLY PERMITTED IN PRODUCTION
- NON-PRODUCTION ONLY

WHAT IT MEANS

Vibe coding is not a side practice, a personal-productivity hack, or a sandbox technique. In the typical surveyed organization, it is written into the software development life cycle (SDLC). AI-generated code is hitting the same production environments, the same revenue-bearing services, and the same customer-facing endpoints as code written by senior engineers, under the same SLAs and the same outage clock.

Once vibe coding sits inside the production policy, every governance system around it has to apply equally regardless of authorship: code review, change management, deployment guardrails, incident response, and audit logging.

The data later in this report shows that most organizations have authorized the practice without consistently leveling up the surrounding controls.

Leaders who close that gap early, by extending the same release health, change tracking, and SLO monitoring discipline to AI-generated code that they already apply to human-authored code, will absorb the operational shock more smoothly than those who treat vibe-coded services as a separate category.

AI-generated code grades higher than human-authored code

61% of leaders rate AI-generated code as "somewhat higher quality" than human-authored code. Another 33% rate it "much higher."

PERCEIVED QUALITY OF AI-GENERATED CODE VS HUMAN AUTHORED CODE

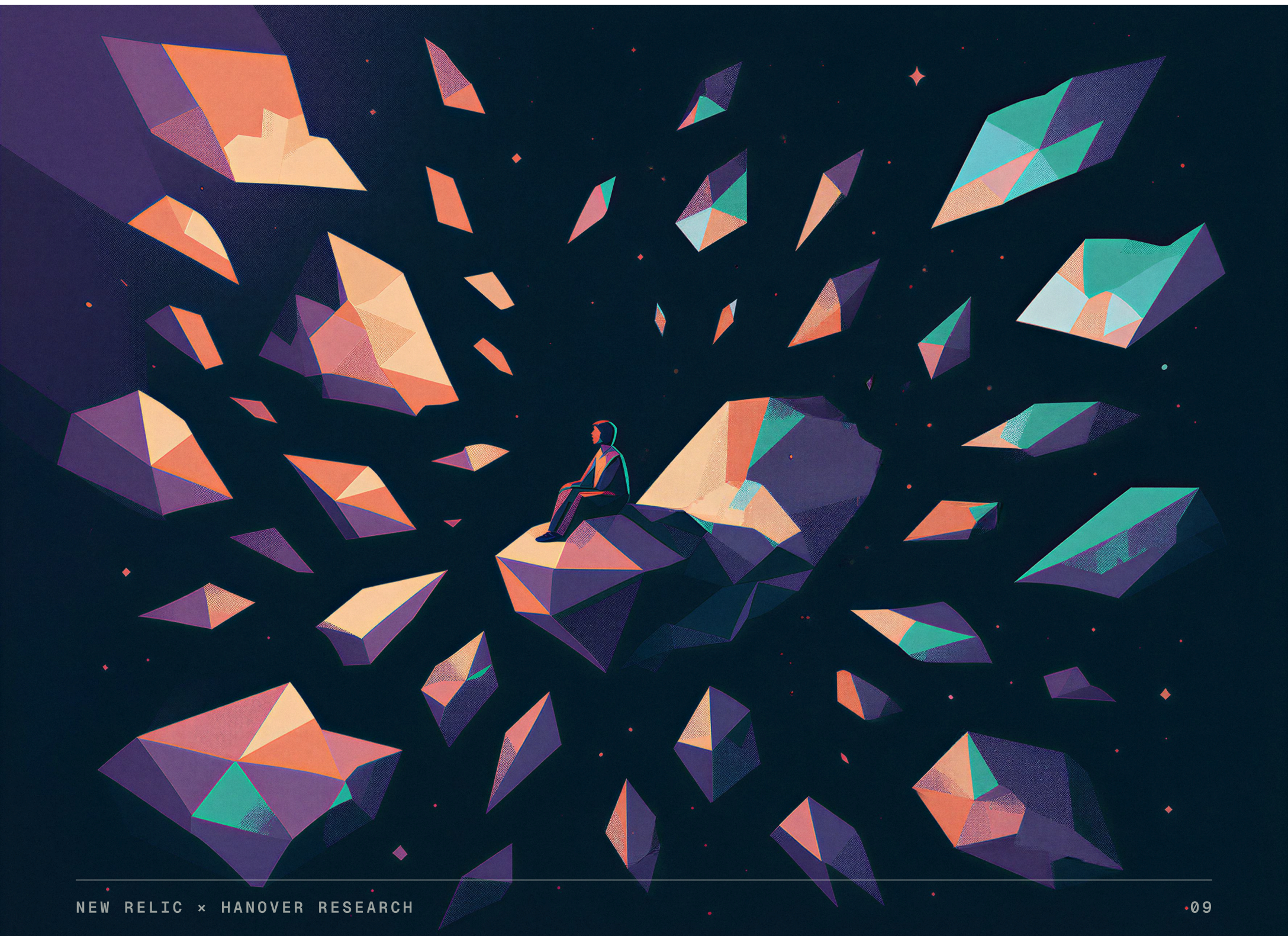
WHAT IT MEANS

The headline benefit of AI-assisted coding is delivering on its promise at the point where most engineering organizations measure quality: code review. Cleaner patterns, consistent style, fewer obvious bugs at submission time. That perception is genuine, it is widespread, and it is part of the reason organizations are willing to keep accelerating vibe coding adoption.

The important caveat is that the quality assessment in this data point is a judgment made at code-review time, not at incident-response time. Code that reads well is not the same as code that operates well, and the metrics that track each of those things look very different. The data across this survey shows what happens once the same code starts running under real traffic, against real dependencies, with real customers on the other end.

THE RUNTIME REALITY

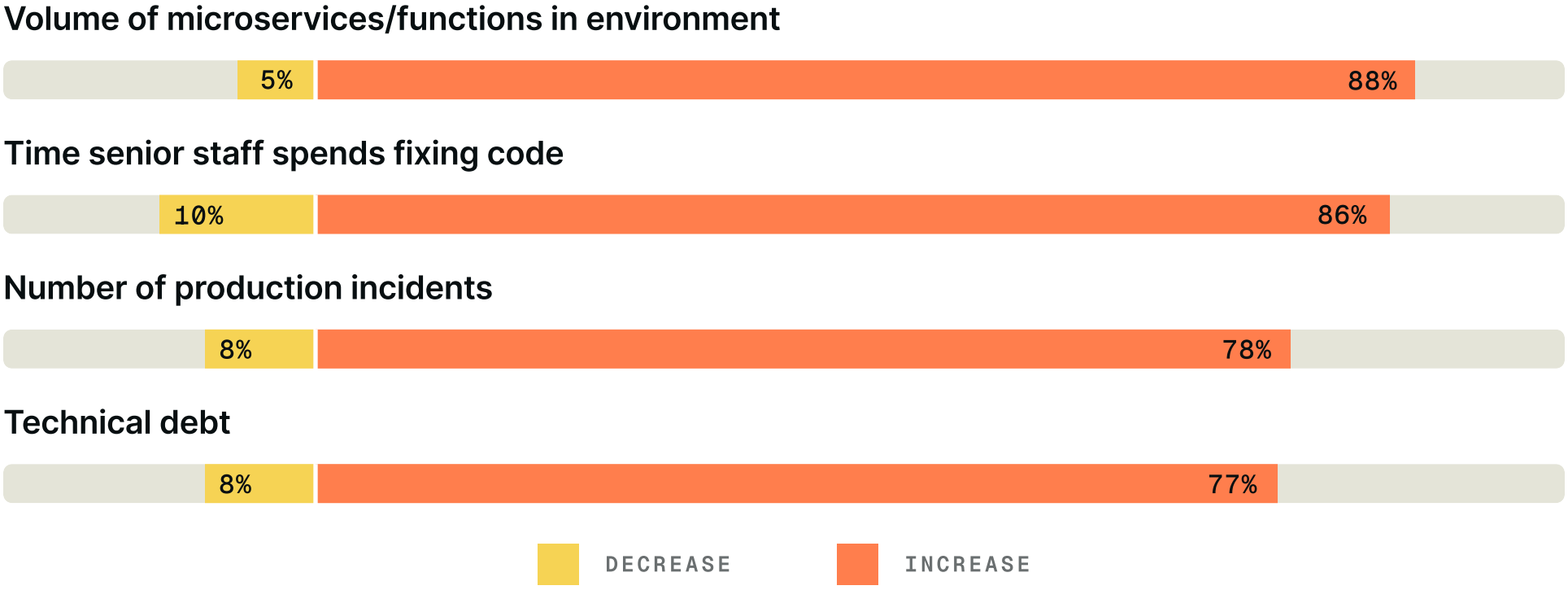
AI code has accelerated in output, but maintaining system reliability has become more expensive and labor-intensive.



The operational tax has risen in lockstep

Senior-engineer rework time has grown at 86% of organizations over the past 12 months. Across the same window, microservice sprawl, production incidents, and technical debt have all risen at three-quarters of organizations or more.

SHARE OF AI-GENERATED CODE REQUIRING SIGNIFICANT REWORK N=200



WHAT IT MEANS

The four metrics above are the most common downstream costs of AI-assisted development, and they have moved in unison over the same 12-month window in which velocity and perceived quality improved. Speed got cheaper. Reliability did not. The same organizations reporting the velocity gains are the ones absorbing the operational load shown here.

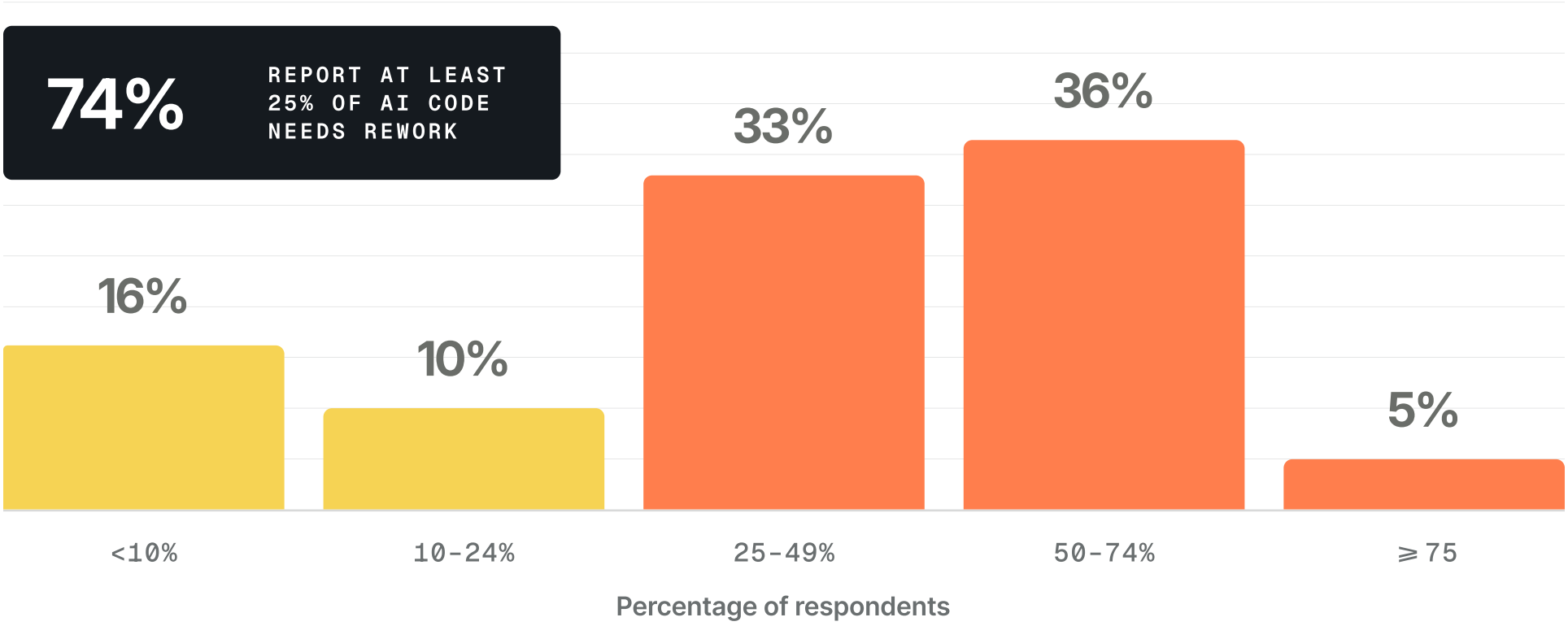
This is the central trade-off of the AI-assisted era, and it is now documented in the data. The right strategic response is not to slow down. The velocity gains are real and revenue reflects that.

Based on increased feature delivery speed, 63% report a modest increase in revenue. The right response is to budget explicitly for the operational overhead that comes with them. That means accounting for incident rates, MTTR, technical debt service, and senior-engineer rework time in the same capacity plan that targets deployment frequency and feature throughput. Organizations that publish both sides of the ledger to their executive teams will make better decisions about where to push velocity further and where to pause and reinforce.

AI-generated code is leading to an uptick in rework

41% of leaders say at least half of their AI-generated code requires significant rework due to poor context, incomplete data, or incorrect assumptions. 74% say at least a quarter does.

SHARE OF AI-GENERATED CODE REQUIRING SIGNIFICANT REWORK N=200



WHAT IT MEANS

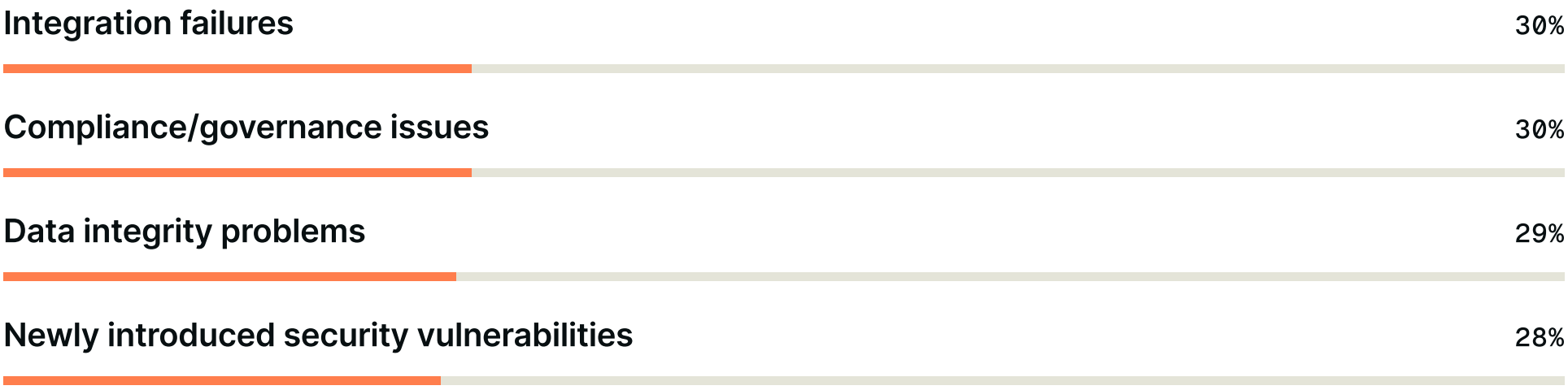
The “fix-forward” cost of vibe coding is substantial. Even in organizations where AI is producing the majority of code, a meaningful share of that output is going back for repair, not for polish but for missed context, hallucinated interfaces, or wrong assumptions about data. This rework is increasingly being done by senior engineers who would otherwise be working on higher-leverage problems.

The root cause is that AI coding tools generate code without seeing how the system actually behaves at runtime. They can read the source, but they cannot read the trace. The organizations that have started closing this gap (by piping production telemetry, real query patterns, and real dependency graphs back into the development loop) are the ones reporting the lowest rework rates. Closing the loop between runtime data and code generation is likely to be the most consequential observability practice of the next 24 months, and it is the practice that most directly reduces the size of the operational tax.

Organizations are feeling the impact of failure modes

Just 19% of organizations report no AI-generated code challenges in the last six months. The other 82% each cite an average of more than two production issues.

TOP PRODUCTION CHALLENGES FROM AI-GENERATED CODE IN THE PAST 6 MONTHS N=200



WHAT IT MEANS

Four distinct failure modes (integration failures 31%, compliance/governance issues 30%, data integrity problems 29%, and newly introduced security vulnerabilities 28%) have each affected roughly three in ten organizations in a six-month window. Historically, human-authored code that's peer reviewed has established stable baseline failure rates, but AI-generated code introduces roughly 1.7 times more critical runtime issues. AI-generated code is not breaking production in one signature way. It is breaking it in many small ways simultaneously, and the breakage is widespread enough that most organizations have at least one war story from the last two quarters.

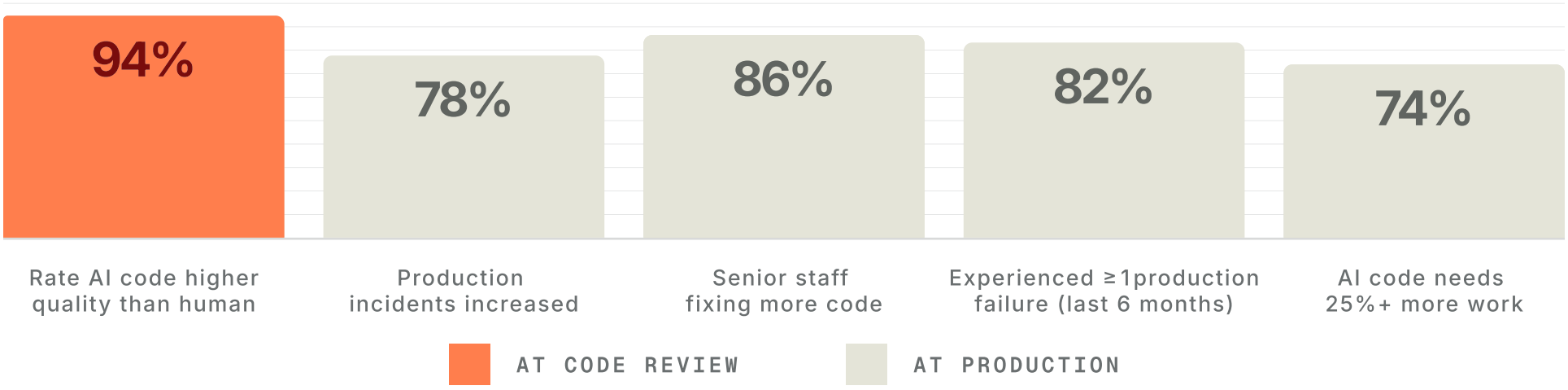
Each of the four named failure modes has a distinct observable signature.

Integration failures show up as schema drift, contract violations, and rising error rates between services. Compliance issues surface in audit logs and policy violations. Data integrity problems appear as anomalies in production telemetry: duplication, drift, unexpected null rates. Security vulnerabilities expose themselves in trace data, authentication patterns, and behavioral anomalies. Organizations that have mapped each failure mode to a specific detection pattern in their observability platform will close the gap between "an AI-generated mistake reached production" and "we caught it before it caused real damage" much faster than those still relying on user complaints or scheduled audits. This is the practical case for a unified telemetry surface, and it is one of the reasons New Relic, and platforms like it, are seeing rising AI-code workloads.

The contradiction at the center of the AI-assisted era

The same organizations that rate AI-generated code higher quality than human-authored code at review time (94%) are reporting more incidents (78%), more rework (74% say 25%+ of code needs rework), more senior-engineer firefighting (86%), and more production failures (82% in the past six months) from that same code once it ships.

AI CODE LOOKS BETTER IN REVIEW, PERFORMS WORSE IN PRODUCTION N=200



WHAT IT MEANS

This is the most important pattern in the study. Reviewers see code that looks cleaner, more consistent, and better-structured than what humans typically produce. Production sees code that fails more often, in more ways, and requires more senior-engineer attention to fix. Both pictures are accurate. But, they are not measuring the same thing. The first measures readability and surface quality. The second measures behavior under real load, real dependencies, and real edge cases—the conditions AI coding tools have no visibility into when they generate the code.

The contradiction has a single resolution: The reason AI-generated code looks good in review and performs worse in production is that the AI is working with the artifacts a human reviewer can see (the source) and is blind to the artifacts only production can produce (the trace).

Closing that gap is the strategic job of the observability layer for the next several years. The organizations that win the AI-assisted era will be the ones that treat their production telemetry as a first-class input to the development loop, not just an after-the-fact debugging tool; a feedback signal that improves the next prompt, the next code review, and the next deployment decision. This is the work the observability category exists to do, and the data in this study is the clearest articulation of why it matters.

Blind trust in AI generation is driving a production crisis

62% of technology leaders report that their engineering teams "often" trust AI-generated code enough to ship it to production without line-by-line manual verification.

RESPONDENTS WHO TRUST AI CODE WITHOUT LINE-BY-LINE VERTIFICATION N=200

WHAT IT MEANS

The core contradiction of the vibe coding era may stem from a profound misplacement of trust early in the development lifecycle. This widespread acceptance may explain why perceived code quality remains artificially high during initial reviews, yet crumbles immediately in the wild.

LLMs excel at generating code that works perfectly under isolated, optimal conditions. However, they routinely fail to account for edge cases, concurrency blind spots, outdated API deprecations, and complex state changes.

When shipped unvetted, these structural omissions hide in plain sight until hit with live user traffic.

Passing AI-generated code through to production without inspection can also trigger a massive uptick in localized production incidents. SRE and DevOps teams are increasingly forced to act as "AI code janitors," diverting up to a third of their active work week away from roadmap acceleration just to triage and refactor unvetted machine output.

THE FUTURE FRAMEWORK

Technology leaders universally agree that observability is an absolute necessity when managing AI-generated code.

Observability is now table stakes

39% of technology leaders rate observability "extremely important" when working with AI-generated code, the single most-chosen response. Another 57% rate it as "very important." Not a single respondent rates it slightly or not important.

IMPORTANCE OF OBSERVABILITY WHEN WORKING WITH AI-GENERATED CODE N=200

WHAT IT MEANS

The debate about whether observability is essential for AI-generated code is effectively over. With zero respondents at the bottom two scale points, the question is no longer whether to invest in observability, but rather which platform to standardize on.

When importance scores cluster this tightly at the top, the strategic action shifts from buying to consolidating. Most organizations in this sample already operate an observability stack, and many run more than one tool.

The leaders who get the most leverage from a near-unanimous consensus are the ones who use it to standardize on a single platform with breadth across observability and AIOps, rather than continuing to pay for fragmented coverage that none of the on-call engineers fully trust at 2 a.m.

Platforms like New Relic that unify all four signals under one data model and one query surface are positioned for that consolidation moment.

Telemetry is being written into the AI's prompt

56% of leaders report their teams "often" prompt AI to include specific telemetry (logs, traces, custom metrics) when generating code. Another 22% do so "always." None do so rarely or never.

HOW OFTEN TEAMS PROMPT AI TO INCLUDE TELEMETRY IN GENERATED CODE N=200

78% SAY THEIR TEAMS OFTEN OR ALWAYS PROMPT FOR LOGS, SPANS, AND CUSTOM METRICS

WHAT IT MEANS

Engineers are not waiting until after the code is written to think about observability. They are pulling it forward into the prompt itself. Telemetry is becoming part of the AI's definition of "done."

The choice of what to log, what to trace, and what to alert on is moving upstream, out of the SRE backlog and into the developer's prompt. Two implications follow. One, the quality of an organization's telemetry now depends heavily on the prompts and templates its engineers use.

Two, organizations that standardize on opinionated, schema-aware conventions (OpenTelemetry semantic conventions, structured logging, consistent span attributes) will see a much sharper signal-to-noise ratio than those that let each developer prompt their own format. Whoever sets the standard inside Cursor, Copilot, Claude Code, and ChatGPT effectively sets the observability default for the next phase of this category.

Conclusion

Three findings frame the rest of 2026 for engineering organizations.

VELOCITY IS REAL AND IT IS BEING PAID FOR AT RUNTIME

Faster releases, more frequent deployments, and higher review-time quality are all delivering as promised. The cost is showing up in incidents, rework, sprawl, and senior-engineer time. Leaders who plan for both sides of that ledger will make better calls about where to push and where to reinforce.

THE CONCERN-TO-CONTROL GAP IS THE MOST ACTIONABLE FINDING IN THE STUDY

Leaders are correctly worried about the risks of AI-generated code. The controls they have actually deployed do not yet match the concern. The organizations that close the gap fastest—with layered safeguards across security scanning, deployment guardrails, governance policy, and runtime observability—will be the ones operating AI-heavy codebases at enterprise SLAs without unacceptable risk.

THE CONTRADICTION BETWEEN REVIEW-TIME QUALITY AND PRODUCTION REALITY IS WHERE OBSERVABILITY EARNS ITS PLACE

AI coding tools see the source. They do not see the trace. Until that loop closes, AI-generated code will continue to grade well in review and surprise teams in production. The observability platform that becomes the system of record for what AI code is actually doing, and feeds that signal back into the development loop, sits at the center of how engineering organizations operate in the AI-assisted era. New Relic was built around exactly that loop, which is why this is the layer worth investing in first.

New Relic published this study because the questions it answers are the questions every engineering organization is now wrestling with. The data shows that the era of AI-generated software is here, that its operational consequences are real, and that observability has become the layer where those consequences are managed.

ABOUT HANOVER

Hanover Research is a global market research and analytics firm operating on a fixed-fee subscription model to provide custom quantitative and qualitative insights. Founded in 2003, the organization employs over 200 analysts, statisticians, and survey methodologists. Hanover Research executes primary market research, predictive data modeling, and segment analysis to support strategic, operational, and financial decisions across the corporate, healthcare, and education sectors.

ABOUT NEW RELIC

The New Relic Intelligent Observability Platform ingests, normalizes, and analyzes telemetry data across the software stack. By unifying metrics, events, logs, and traces within a single telemetry data platform, New Relic shifts system engineering from reactive debugging to predictive performance management. That's why organizations including Adidas Runtastic, American Red Cross, Domino's, GoTo Group, Ryanair, Topgolf, and William Hill utilize New Relic to monitor infrastructure health and optimize application performance.

ABOUT THIS REPORT

The data in this report comes from a 2026 online survey conducted by Hanover Research and commissioned by New Relic. The sample includes 200 full-time, U.S.-based technology decision-makers in IT and engineering roles at organizations utilizing generative AI for software development.